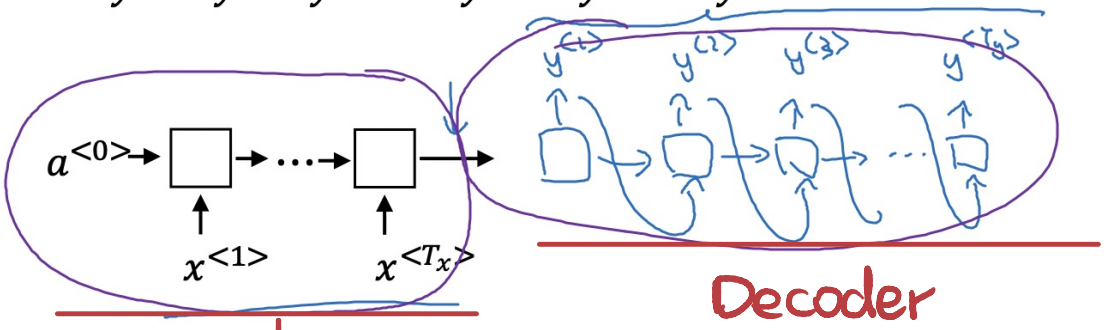
Basic Models

Sequence to sequence model :

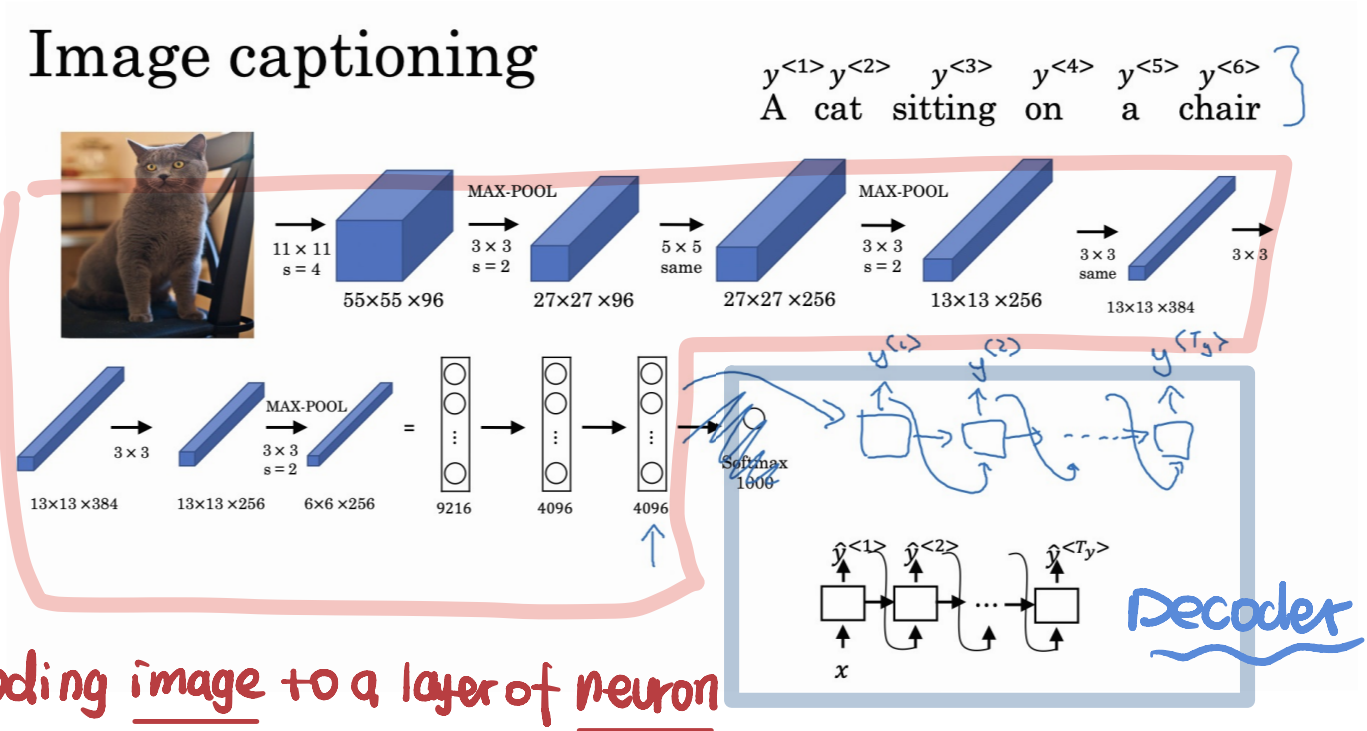
$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ $x^{<4>}$ $x^{<5>}$
Jane visite l'Afrique en septembre

⟶ Jane is visiting Africa in September.
$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$



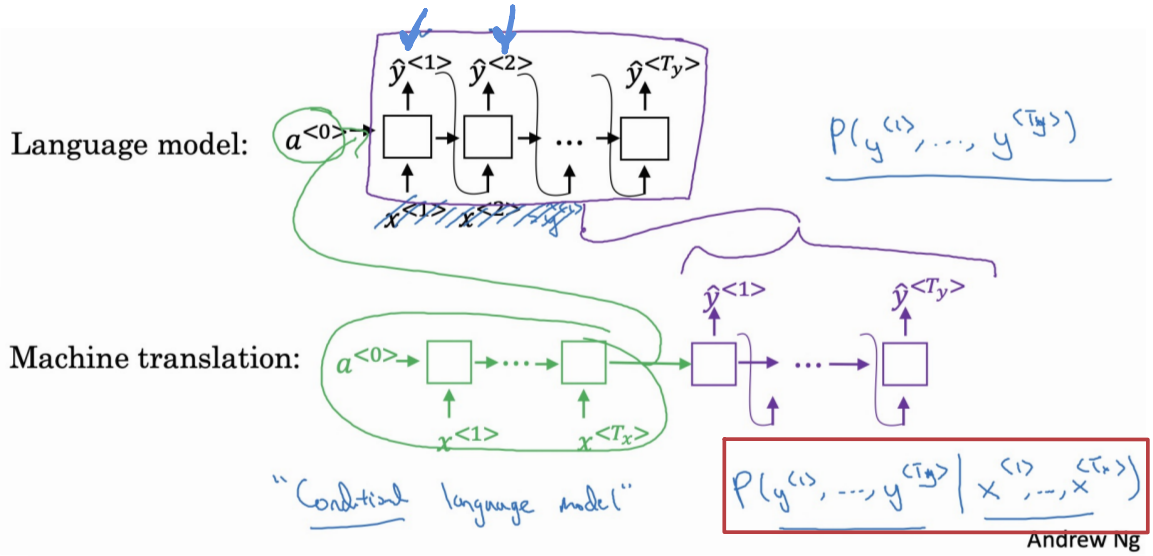
(feed in with $x^{<1>} \to x^{<T_x>}$)

same encoder-decoder technique can be used to create **Image Captioning Model**.



Encoding image to a layer of neuron

Picking the Most Likely Sentence



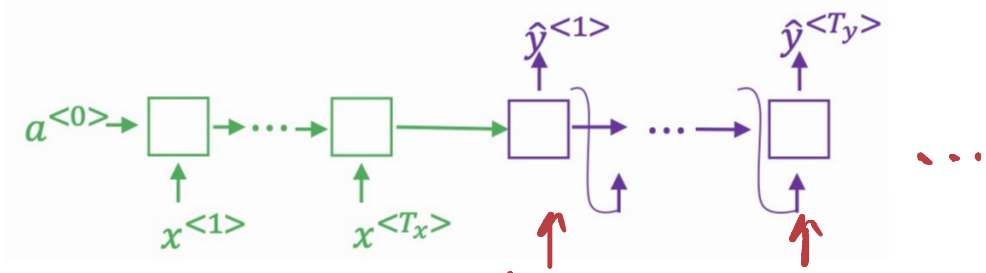
Conditioning Probability given x

Another difference:

With language model, we want to generate output randomly from the predicted distribution

In machine translation: we generate best translation (most likely translation)

Naive Approach: greedy search



Problem: → 1) Jane is visiting Africa in September

→ 2) Jane is going to be visiting Africa in September

Even though sentence 1) is a better translation. Since "going" is used more frequently used in human text, sentence 2) is more likely to be output by greedy

→ Better Solution

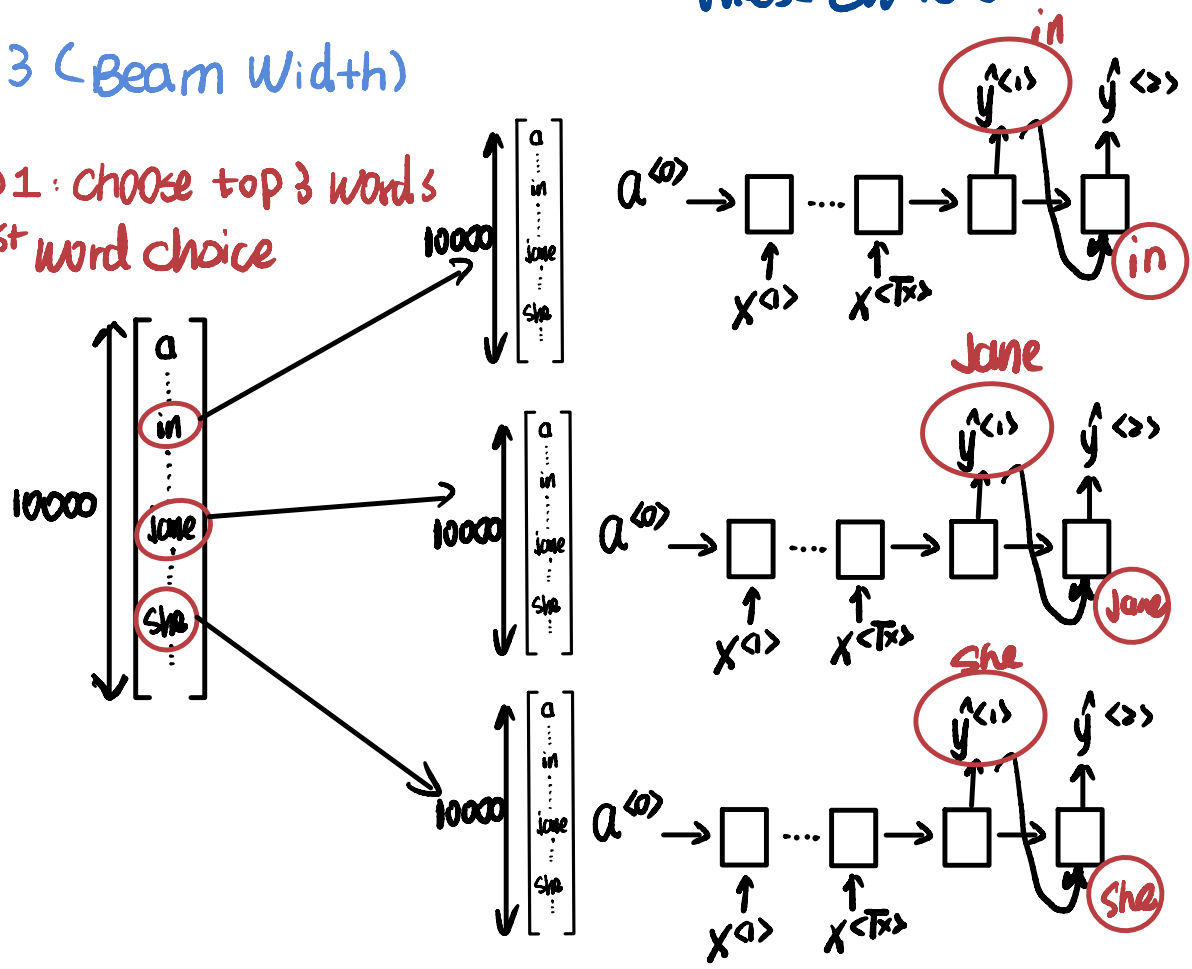
Beam Search

Step 2: Combine each choice from Step 1 with whole Vocabulary to form 3 × 10,000 = 30,000 choices, choose top 3 = Beam width from these choices

B = 3 (Beam Width)

3 RNN concurrently

Step 1: choose top 3 words as 1st word choice



→ $P(y^{\langle 1 \rangle}, y^{\langle 2 \rangle} | x)$

$= P(y^{\langle 1 \rangle} | x) P(y^{\langle 2 \rangle} | x, y^{\langle 1 \rangle})$

length Normalization

$$\arg\max_{y} \prod_{t=1}^{T_y} P(y^{\langle t \rangle} | x, y^{\langle 1 \rangle}, y^{\langle 2 \rangle} \dots y^{\langle t-1 \rangle})$$

Not very numerically stable, prone to underflow, since we are multiplying many <1 numbers.

instead, we use: $\arg\max_{y} \sum_{y=1}^{T_y} \log P(y^{\langle t \rangle} | x, y^{\langle 1 \rangle} \dots y^{\langle t-1 \rangle})$

$$\frac{1}{T_y^{\alpha}} \sum_{y=1}^{T_y} \log P(y^{<t>} \mid x, y^{<1>} \dots y^{<t-1>})$$

$\alpha = 0.7, 1, \dots$ hyperparameter

weighting out the undesirable negative effect of low posibility assigned to relatively long sentences.

Beam Width B?

large B: better result, slower

small B: worse result, faster.

B=10 common

100, production

1000 -> 3000 theoretical papers really big.

- Unlike exact search algorithms, such as BFS and DFS, Beam Search runs faster but is not garanteed to find exact maximum for $\arg\max_y P(y \mid x)$

Error Analysis on Beam Search

Error analysis on beam search

$P(y^* \mid x)$, $P(\hat{y} \mid x)$ → computed third forward propagation through RNN

Human: Jane visits Africa in September. (y^*)

Algorithm: Jane visited Africa last September. ($\hat{y}$)

Case 1: $P(y^* \mid x) > P(\hat{y} \mid x)$ ←

$\arg\max_y P(y \mid x)$

Beam search chose $\hat{y}$. But y^* attains higher $P(y|x)$.

Conclusion: Beam search is at fault.

Case 2: $P(y^* \mid x) \leq P(\hat{y} \mid x)$ ←

y^* is a better translation than $\hat{y}$. But RNN predicted $P(y^*|x) < P(\hat{y}|x)$.

Conclusion: RNN model is at fault.

Error Analysis process: Pick multiple false preds, Perform the above analysis → what fraction of error → beam / RNN model. Andrew Ng

Bleu Score

$$\underset{\text{Bleu Score on unigram}}{\underbrace{P_1}} = \frac{\sum_{\text{unigram} \in \hat{y}} \text{Count}_{clip}(\text{Unigram})}{\sum_{\text{unigram} \in \hat{y}} \text{Count}(\text{Unigram})}$$

$$P_2 = \frac{\sum_{\text{ngram} \in \hat{y}} \text{Count}_{clip}(\text{ngram})}{\sum_{\text{ngram} \in \hat{y}} \text{Count}(\text{ngram})}$$

P_n = Blue score on n-grams only $\quad P_1, P_2, P_3, P_4$

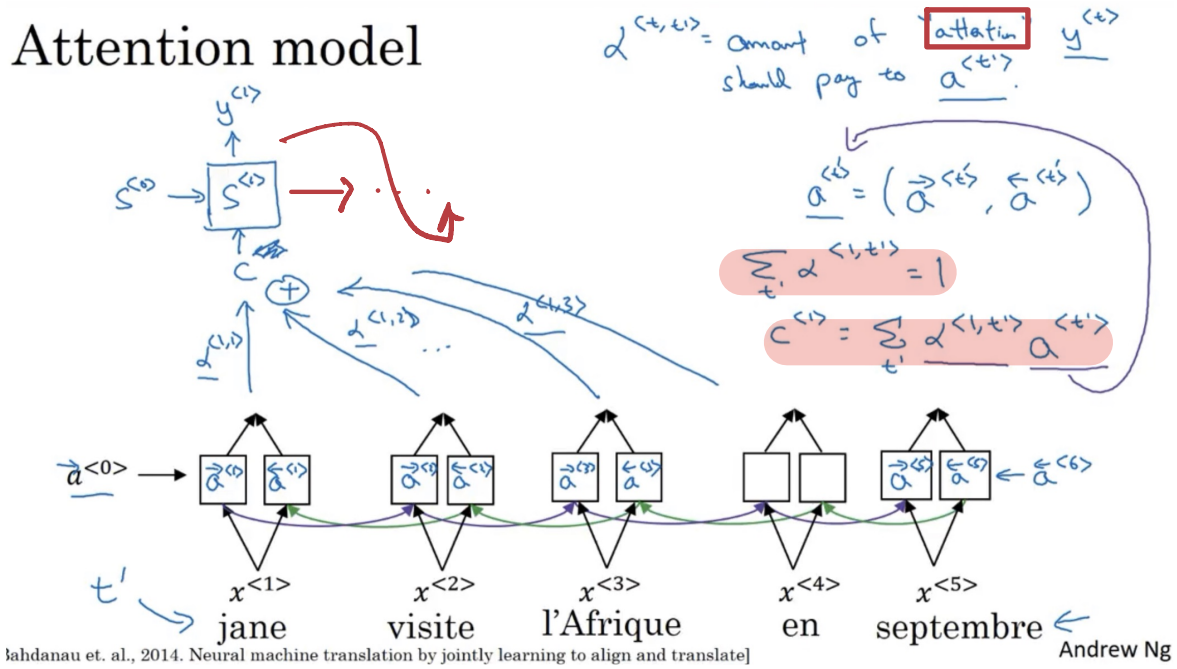
Combined Bleu score : $\underset{\text{brevity penalty}}{\underbrace{BP}} \exp\left(\frac{1}{4}\sum_{n=1}^{4} P_n\right)$

$$BP = \begin{cases} 1 & \text{if MT\_output\_length} > \text{reference\_length} \\ \exp(1 - \text{ref\_length} / \text{MT\_output\_length}) & \end{cases}$$

Attention Model

Intuition: When human translates, we won't read whole sentence and output entire translation in one iteration. Rather, we read in and output translation part by part at the same time.

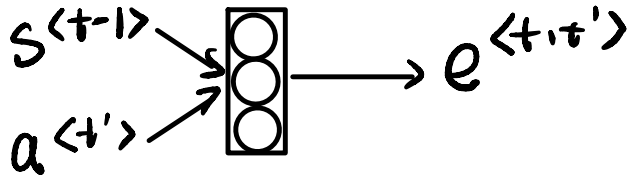
Similarly, neural networks are also bad at processing large sequence of input at once. Thus we designed attention model which can figure out how much attention we should pay to nearby words when translating current word.



Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

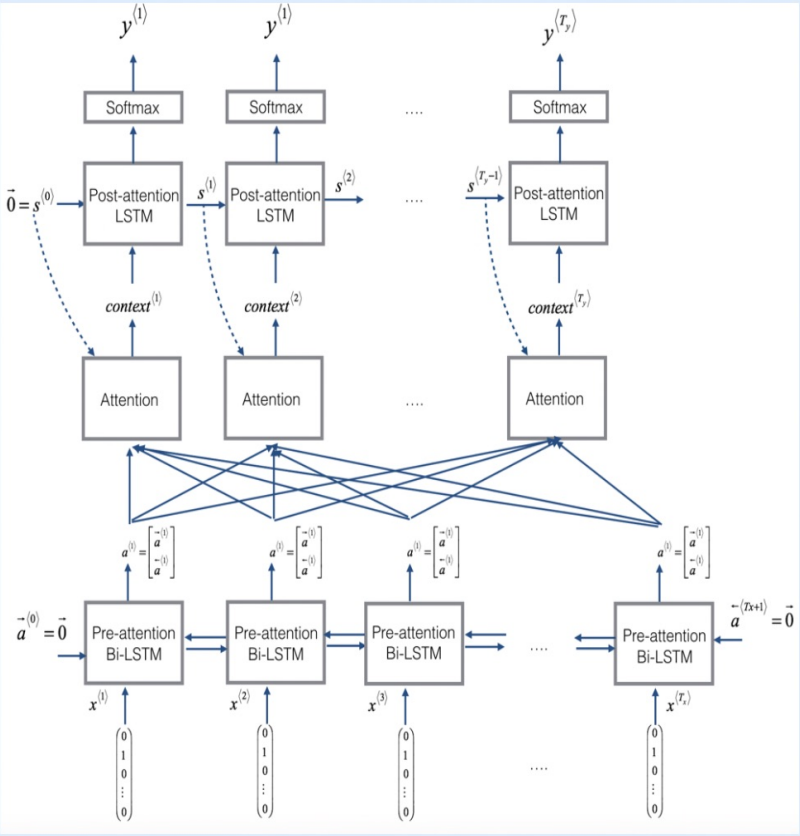
$a^{\langle t,t' \rangle}$ = amount of attention $y^{\langle t \rangle}$ should pay to $a^{\langle t' \rangle}$

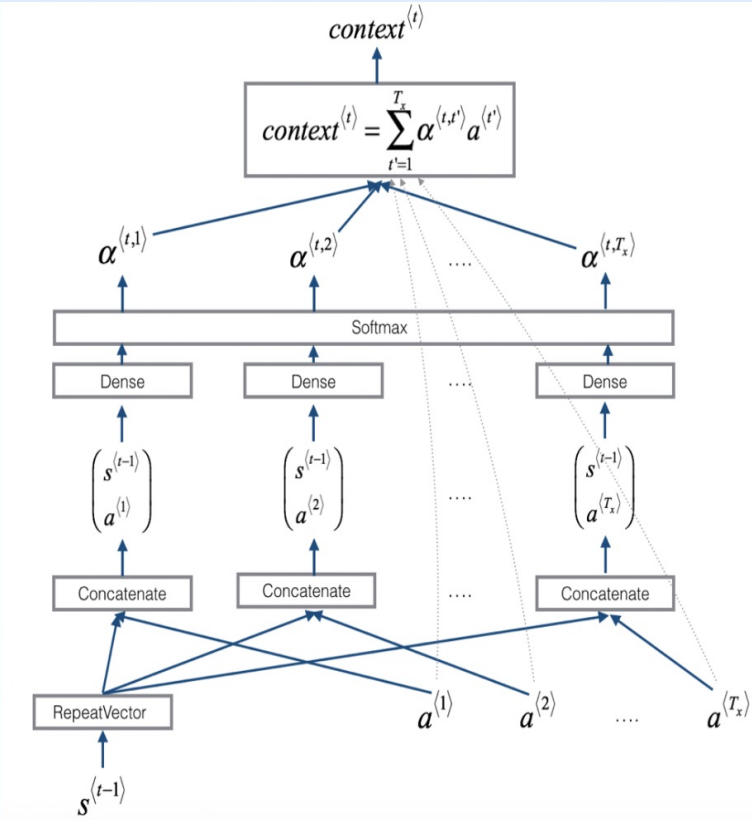
$$a^{\langle t,t' \rangle} = \frac{\exp(e^{\langle t,t' \rangle})}{\sum_{t'=1}^{T_x} \exp(e^{\langle t,t' \rangle})}$$



Attention depends on last translation and coming hidden activation from encoder.

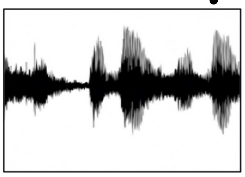
- The diagram on the left shows the attention model.
- The diagram on the right shows what one "attention" step does to calculate the attention variables $\alpha^{\langle t,t' \rangle}$.
- The attention variables $\alpha^{\langle t,t' \rangle}$ are used to compute the context variable $context^{\langle t \rangle}$ for each timestep in the output ($t = 1, \ldots, T_y$).



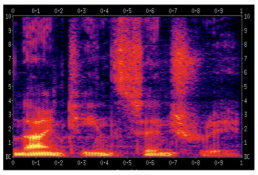


Speech Recognition

x: audio clip → y: transcript

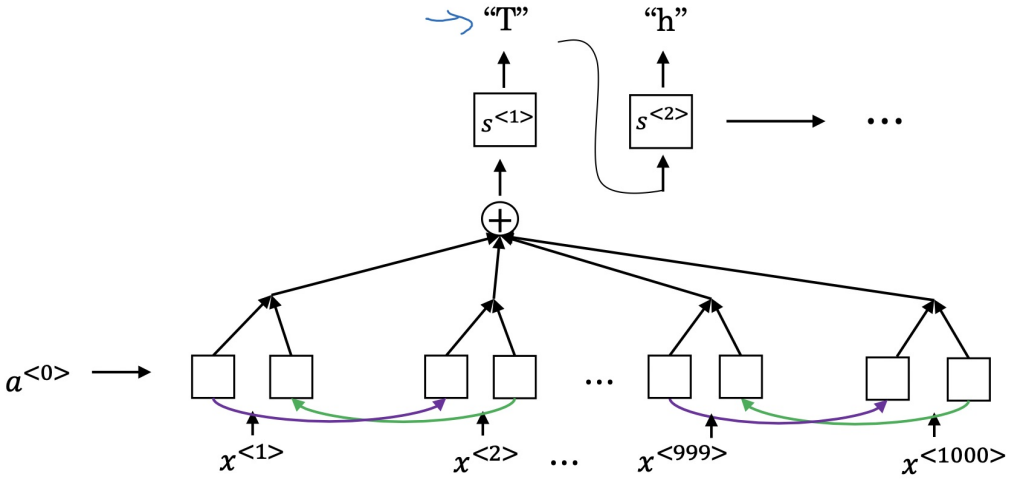


↓ possible preprocessing



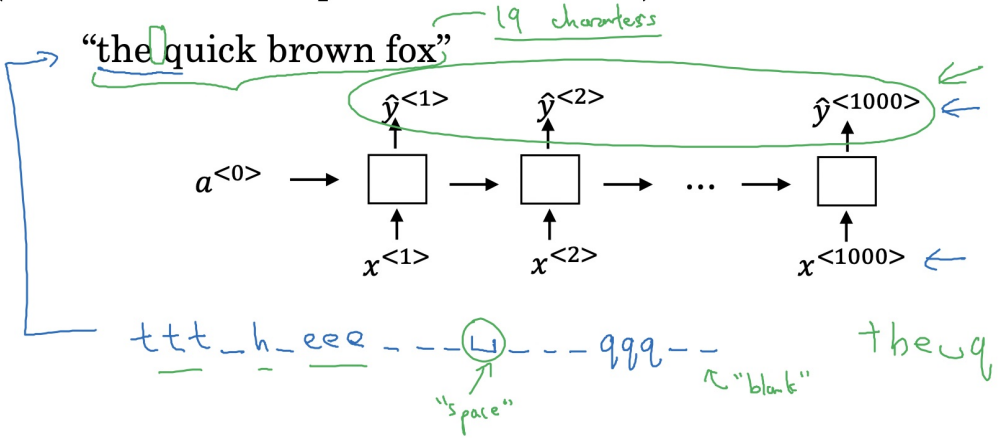
(spectrograms)

Attention model for speech recognition



CTC cost for speech recognition

(Connectionist temporal classification)



Basic rule: collapse repeated characters not separated by "blank"

[Graves et al., 2006. Connectionist Temporal Classification: Labeling unsegmented sequence data with recurrent neural networks]

Trigger Word Detection